Before The Board of Patent Appeals and Interferences

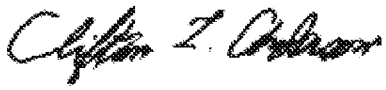
For: **Computer Cluster With Second-Node Instance of Application Having Access to State Snapshot of First-Node Instance of Application**			
Applicant: Kenneth PRIDDY et al.		Attorney Docket No.: 200314321-1	
Serial No.: 10/806,261 (1933)	Filed: 2003-Mar-31	Art Unit: 2192	Examiner: Amine RIAD

Director For Patents
PO Box 1450
Alexandria, VA 22313-1450

Reply Brief

This Reply Brief is in response to the Examiner's Answer mailed 2008-May-05.

Respectfully submitted
Kenneth PRIDDY et al.
by



Clifton L. Anderson
Reg. No. 30,989
(408) 257-6070

TABLE OF CONTENTS

	Signature Page	RB-1
	Table of Contents	RB-2
	Reply-Introduction	RB-3
	Reply-Overview	RB-3
	Reply-Remap	RB-5
	Reply to Rejection of Group 1	RB-6
	Reply to Rejection of Groups 2-7	RB-14
	Reply to Examiner's Response Group 1	RB-14
	Reply to Examiner's Response Group 2	RB-20
	Reply to Examiner's Response Group 3	RB-22
	Reply to Examiner's Response Group 4	RB-23
	Reply to Examiner's Response Group 5	RB-23
	Reply to Examiner's Response Group 6 and 7	RB-24
	Conclusion	RB-25

REPLY

[01] INTRODUCTION

[02] Appellants concur with Items 1-8 and 11 of the Examiner's Answer. This Reply is directed primarily to Item 9 (Grounds of Rejection) and Item 10 (Response to Arguments). Because Appellants' remarks regarding these items is rather detailed, the following overview is presented to focus on a primary issue.

[03] OVERVIEW

[04] What the present invention and Kelkar have in common is the prospective copying of data on a first node to a backup node so that the data is available to the backup node in case the first node fails. However, where the present invention deals with continuously changing application state data, Kelkar deals with infrequently changing resource configuration (definition) data.

[05] The problem with copying data that is being continuously modified is that the data changes as the copy is made. As a result, a portion of the data copied early in the copy process may not be coherent with data copied late in the copy process. A "snapshot" is a coherent copy of data as it exists at a given moment in time. To take a snapshot, data modification may have to be interrupted; the present invention minimizes this interruption by creating the snapshot in relatively fast volatile memory rather in than the relatively slow memory typically used as storage media. Once the snapshot is made, data modification can resume (so that the data diverges from the state represented by the snapshot) while the snapshot is transferred to any required destination.

[06] In Kelkar, the resource configuration data is changed infrequently, e.g., only when an administrator decides to make a configuration change. There is, in general, sufficient time between resource-configuration updates to complete a configuration change, to update a configuration file to describe the new configuration, and to propagate a description of the new configuration to other computers. To propagate the resource configuration data, a copy must be made and there is nothing to prevent one from calling that copy a "snapshot". However, if the propagation occurs soon after the configuration change, there will be little opportunity for the data that is the subject of the snapshot to diverge before propagation is complete. In any event, Kelkar does not disclose that the original data will or could change while the data is propagated or otherwise transferred, so the limitations in inedependent Claims 1 and 7 to that effect are not met by Kelkar. Accordingly, the rejections for anticipation of these independent claims and the remaining claims that depend form them should be reversed.

[07] Kelkar does not disclose modifying the data copied by the snapshot while the snapshot is being transferred, and the Examiner has not argued otherwise. However, in the Examiner's Answer, near the bottom of page 8, the Examiner has developed a theory that propagating updates in real time implies that data is modified as a snapshot is transferred.

> while the
>
> updates take place Kelkar's system is working in real time or near real time to make sure that all the updates received at **node 110A** level are copied and synchronized along the other nodes. The fact that Kelkar works in real time shows that Kelkar transfers the snapshots while Kelkar is still updating node 110A.

[08] In fact, Kelkar does not confirm the Examiner's theory. Kelkar does not disclose that synchronization occurs in real time, but only as close as possible to real time. Kelkar is explicit that a resource configuration is changed (action 3.1), then the local resource definition is updated (action 3.2), and then the new resource definition is propagated (actions 3.5-3-8), see Kelkar, column 6, lines 5-27. Kelkar does not transfer snapshots while updating a node. Kelkar does not teach, imply, or suggest that the claim requirement that data represented in a snapshot is modified while the snapshot is transferred. Accordingly, the rejections for anticipation of all claims by Kelkar should be reversed.

[09] These and other new statements in the Examiner's Answer are addressed in greater detail below.

"REMAP"

> **NOTE**: The Examiner made a remap of only one physical element of the recited independent claims. The remap is as follow:
>
> Non volatile memory is remapped to item 140d instead of 140 the shared storage drive.

[10] This passage from the Examiner's Answer, page 3, suggests that independent Claims 1 and 7 recite "non volatile memory" and that Kelkar discloses an element referred to as "140d". However, Claims 1 and 7 do not recite "non volatile memory" and Kelkar does not disclose an element "140d", so this statement about a "remap" is unclear.

[11] Independent Claims 1 and 7 both recite "storage media" so that could be what the Examiner refers to as "non volatile memory". Confusingly, the Examiner's Answer annotates the first recitation of "storage media" in Claim 1 as "item 140" and a second recitation as item

"140D". Kelkar discloses "Storage Resource Definition 140D", but this would seem to be data rather than storage media. However, it might be presumed that this data would be stored in some sort of storage media. Taking all this into account, Appellant speculates that the proposed "remap" is of the claimed storage media to some storage media in which "Storage Resource Definition 140D" is stored.

[12] GROUP 1: Claims 1 and 3-5

[13] "Storage Media"

Kelkar discloses a computer cluster comprising:

- storage media; (Figure 1; item 140)

[14] This statement, which begins the Examiner's annotation of Claim 1 and which is found near the bottom of page 3 of the Examiner's Answer, appears to associate the claimed storage media with Kelkar's Storage Resource 140, which no doubt qualifies as a storage media. However, the foregoing "remap" suggests that the claimed storage media was "remapped" to Storage Resource Definition 140D", so confusion remains. Appellant assumes that the Examiner simply forgot to apply the remap to this first instance of "storage media", but it is hard to be certain.

[15] "First Computer"

- a first computer having a first instance of an application program installed, said application program having instructions, (Figure 2; item 110A)

[16] This statement is part of the continued annotation of Claim 1 and appears at the bottom of page 3 of the Examiner's Answer. It is not clear from the statement itself whether "item 110A" is intended to map to the claimed first computer or to the claimed application program or to the instructions. Kelkar refers to item 110A as a "node", which is how computers on a network or in a cluster are often referred to. Accordingly, Appellant assumes that the Examiner is mapping the claimed "first computer" to Kelkar's node 110A. This interpretation is consistent with Fig. 2, which depicts item 110A as a computer tower.

[17] The problem with this interpretation is that it leaves the claimed "application program" unmapped. Another problem is that the Examiner is improperly combining elements of two different systems, a computer system 100 depicted in Fig. 1 and a cluster 220 shown in Fig. 2. Since the Examine has not shown Kelkar discloses the claimed elements in the claimed combination, the rejections for Claims 1 and 3-5 should be reversed.

[18] "Volatile Memory"

said first computer including,

- volatile memory; (Figure 1;item 104A)

[19] This statement is a continuation of the Examiner's annotation of Claim 1 and appears at the top of page 4 of the Examiner's Answer. The

statement purports to map the claimed "volatile memory" to item 104A of Kelkar. Kelkar, column 4, describes item 104A as follows.

> Storage management service **104**A is a service for managing storage resources, such as storage resource **140**, in a multi-vendor storage environment. In the example shown, storage management service **104**A communicates with storage access interface **102**A when a storage resource configuration is to be changed.

[20] So the Examiner is mapping the claimed "volatile memory" to Kelkar's Storage management service 104A. The Examiner's Answer does not explain how a storage management service can servie as volatile memory. Appellants speculate that the Examiner assumes that Kelkar's Storage management service 104A is an application or process that runs in volatile memory. It should emphasized that these are assumptions and that Kelkar does not relate Storage management service 104A in any way to volatile memory. Accordingly, the rejections for anticipation by Kelkar should be reversed.

[21] "Processing Means"

- processing means (Figure 7; item 714)

[22] This statement maps the claimed processing means to Kelkar's item 714, which Keklar describes as a central processor. However, it is a central processor for a computer system 710, not node 110A. So the Examiner has not shown where Kelkar has disclosed the claimed processor of the claimed first computer (110A). Accordingly, the rejection for anticipation by Kelkar should be reversed for this additional reason.

[23] "Modify Data"

> for executing instructions of said first instance of said application program so as to modify data stored in said volatile memory

[24] Note that this statement does not map the claimed "data" to an element in Kelkar, nor has the Examiner mapped the claimed application program to an element in Kelkar. Perhaps the Examiner is treating Kelkar's storage management service 104A as 1) volatile memory, and 2) an application program, and 3) data modified by the application program. However, Kelkar does not disclose that Storage Management Service 104A modifies any data. Instead, Kelkar teaches that storage management service 104A implements commands received using a command-line interface. Since Keklar does not disclose that storage management service 104A modifies data in volatile memory, the rejections for anticipation should be reversed.

[25] "Creating a Snapshot"

> creating a snapshot of said data while said first instance of said application program is running, (Column 4; lines 36-37) and (Column 4; lines 55-56)

[26] Appellants have yet to ascertain what elements of Kelkar correspond 1) to the claimed data, and 2) to the claimed snapshot. Kelkar, column 4, lines 36-37 reads as follows.

> In action 1.1, an update to a storage resource configuration is provided to storage management service 104A.

[27] This passage from Kelkar does not mention a snapshot or data explicitly. Perhaps, the update serves as the data, although Kelkar does not disclose that storage management service 104A modifies the update.

[28] The following passage includes what Appellant's believe the Examiner considers to be Kelkar, column 4, lines 55-56 (but is probably lines 54-55 due to a misregistration in the reference).

> To make resource configuration available to another node that can resume operation of node **110**A upon failure, the invention synchronizes resource configuration data on multiple nodes in a clustering environment.

[29] From this passage, it appears that the Examiner is mapping the claimed data on resource configuration data. The fact that this resource configuration data is synchronized on multiple nodes suggests that there are multiple copies of the resource configuration data. The Examiner may be considering these copies to be snapshots. Again, this is all speculation based on the Examiner's partial mapping of claim elements to disclosed elements.

[30] "Snapshot in Volatile Memory"

- said snapshot being stored in said volatile memory,

[31] Kelkar does not disclose this limitation and the Examiner's Answer does not contain an explanation of the Examiner's position to the contrary.

[32] "Diverge"

> while said first instance of said application continues to modify said data so that it diverges from said snapshot transferring said snapshot from said volatile memory to said storage media (Figure 1; item 140D), (Column 4; lines 44-46)

[33] The claims require that the data represented in the snapshot diverge from the state represented in the snapshot while the snapshot is being transferred from volatile memory to a storage media. In no way does Kelkar disclose this limitation. There is no speculation or liberal reading of Kelkar that corresponds to this limitation.

[34] The Examiner's Answer refers here to Kelkar, Fig. 1, item 140D as the storage media. (Elsewhere, we have pointed out inconsistencies in this mapping.) The issue here is whether Kelkar discloses that resource configuration data is modified while the snapshot is being transferred to item 140D. The Examiner's Answer refers to Kelkar, column 4, lines 44-46, which are represented in the following quote.

> In action **1.2**, the update to the storage resource configuration is provided to storage access interface **102A**, which updates storage resource definition **140**D for storage resource **140** in action **1.3**. In action **1.4**, storage access interface **102A** notifies storage management service **104A** that the configuration update is complete.

[35] It is not clear here or elsewhere, what in this passage corresponds to the application data and what corresponds to the snapshot. If the "update" is the data and the copy of that data being transferred to the

storage resource definition 140D is the snapshot, Kelkar does not disclose modifying the update data during the transfer.

[36] In addition, there is the problem that actions 1.1-1.4 relate to the Kelkar's prior art shown in Kelkar, Fig. 1, whereas the Examiner relies on elements of other prior art (Fig. 2) and embodiments of Kelkar's invention (Figs. 3 and 7) for disclosure of other claim elements. Since the Examiner has not shown the claimed elements in the claimed combination, the rejections should be reversed.

[37] "Second Computer"

- and a second computer having a second instance of said application program installed, (Figure 2; item 110b)

[38] Again, it is hard to determine from the Examiner's Answer whether item 110B refers to an application or to a computer. However, Kelkar refers to item 110B as a node, which suggests a computer. The Examiner's Answer does not map the claimed second instance of an application program to any element disclosed by Kelkar.

[39] In addition, the Examiner is improperly combining elements from different systems. Although Kelkar uses like numbers for like parts across the figures, Figs. 1 and 2 represent two different computer systems. "FIG. 1 is a diagram of a typical system 100 for storage resource configuration" (Kelkar, column 4, lines 8 and 9). Kelkar, Fig. 2, is a block diagram of a cluster 220.

[40] "Access"

> said second computer including means for accessing said storage media so that said second instance of said application can access said snapshot as stored on said storage media (Figure 3 ; item 370B).

[41] The Examiner appears to be mapping the claimed means for accessing item 140D (storage resource definition) to item 370B, which Kelkar describes as Resource Configuration Data. It is not clear how resource configuration data can serve as a means for accessing storage media on another computer.

[42] Once again, the Examiner is combining elements from different systems. Fig. 3 "is a diagram of a clustering environment in which storage configuration changes can be made dynamically and are communicated throughout the cluster" 300. In other words, Kelkar, Fig. 3, illustrates an embodiment of Kelkar's invention, whereas, Kelkar's, Figs, 1 and 2 do not provide for communicating configuration changes through clusters. The Examiner is combining elements from an embodiment of Kelkar's invention with instances of Kelkar's prior art. Since the Examiner has not found the claimed elements in the claimed combination, the rejections for anticipation should be reversed.

[43] GROUPS 2-7: Claims 2 and 6-12

[44] The Detailed Action presented in the Examiner's Answer does not differ materially from that in the final action. Accordingly, Appellant relies on the Appeal Brief to address the rejections of these groups and claims.

[45] EXAMINER'S RESPONSE GROUP 1, Claims 1 and 3-5

[46] Introduction

> **Group 1**
>
> In regard the first arguments which states "Final Action fails to establish that Kelkar discloses the claim limitation of creating a snapshot of application data in volatile memory, the rejection for anticipation of claim 1 by Kelkar should be reversed", "Final Action fails to establish that Kelkar discloses the claim 1 limitation of storing a snapshot in volatile memory", and "Final Action fails to establish that Kelkar discloses the claim 1 limitation of transferring a snapshot from volatile memory to storage media". Examiner respectfully disagrees. Examiner points Appellant into two parts of the reference.

[47] These statements, made at the end of page 7 to the beginning of page 8 of the Examiner's Answer, purport to introduce a two-part counter-argument by the Examiner.

[48] Part 1: "Update"

> First, figure 1, this figure shows 3 elements 104A, 102A, and 140D all in connection with storage resource 140. The same figure shows 4 steps 1.1, 1.2, 1.3, 1.4, with transitional arrows as depicted in the figure. Following the description of figure 1 as disclosed in (Column 3), Examiner concludes that **node 110A** receives update of storage resource configuration at step 1.1 at element 104A, the update passes through step 1.2 from element 104a to element 102A, it then finally settles in item 140D by using step 1.3. During this transition it is well apparent that the itinerary of the update goes from volatile memory 104A to non volatile memory 140D, and that is how Kelkar meets the limitation of volatile, and non-volatile memory.

[49] The foregoing is the entire first part of the Examiner's response to Appellants' assertions that the Examiner has not established that Kelkar discloses snapshot-related limitations. Yet, this first part does not even mention a "snapshot".

[50] The foregoing passage from the Examiner's Answer does mention a snapshot or an update that is transferred from what the Examiner considers volatile memory (storage management service 104A) to what the Examiner considers non-volatile memory (storage resource definition 140D). The Examiner does not indicate whether the update is mapped to the claimed snapshot or to the data from which the snapshot is generated, or some combination of the two. In any event, it is unreasonable that Appellants should have to speculate this late in prosecution as what prior-art elements correspond to claim elements.

[51] Furthermore, the Examiner's Answer provides a misleading view of the disclosed updates. The Examiner interprets "update:" to be a thing that passes from one destination to another, finally "settling" at item 140D. This is not what Kelkar discloses in the column 4 section referred to above regarding Kelkar's actions 1.1-1,4.

> In action **1.1**, an update to a storage resource configuration is provided to storage management service **104**A. Typically, an update to a resource configuration is made by an administrator using a command line interface (not shown), but other ways of making resource configuration changes are also within the scope of the invention. In action **1.2**, the update to the storage resource configuration is provided to storage access interface **102**A, which updates storage resource definition **140**D for storage resource **140** in action **1.3**. In action **1.4**, storage access interface **102**A notifies storage management service **104**A that the configuration update is complete.

[52] Kelkar discloses that the storage access interface 102A updates storage resource definition 140D. This is not the same as an update "passing through" storage access interface 102A to storage resource definition 140D. Kelkar does not disclose that storage resource definition 140D (which the Examiner considers to be storage media) receives a snapshot or other copy of the update as received by storage management service 104A (which the Examiner considers to be volatile memory). Kelkar does not disclose that an update or anything else passes from storage management service 104A to storage resource definition 140D. Rather, Kelkar teaches that the results of the change implemented by storage management service 104A is reflected in an updated storage resource definition 140D.

[53] Part 2: Synchronization

> Second, In regard the creation of a snapshot, Kelkar discloses (in Column 4 lines 49-55 and Column 3 lines 33-36), to avoid the problem of accessing resource configuration when node 110A fails, Kelkar synchronizes resource configuration data on multiple nodes, this means while the updates take place Kelkar's system is working in real time or near real time to make sure that all the updates received at **node 110A** level are copied and synchronized along the other nodes. The fact that Kelkar works in real time shows that Kelkar transfers the snapshots while Kelkar is still updating node 110A. The arguments are not correct.

[54] The Examiner has not been able to find a disclosure in Kelkar to the effect that data changes as the configuration data is synchronized. In lieu of an actual disclosure, the Examiner presents his theory that real time synchronization involves transferring a snapshot as a configuration change is being made. The disclosure from Kelkar, column 3, cited by the Examiner above, is included in the quote below, which teaches that synchronization occurs "in as close to real-time as possible", which is not the same as real time.

> The present invention provides a method, system, and computer program product to make resource configuration information available to nodes in a cluster in as close to real-time as possible with minimal overhead. Resource configuration data are synchronized at nodes in the cluster, thereby enabling a node having synchronized configuration data to resume operations of a failed node.

[55] The disclosure from Kelkar, column 4, cited by the Examiner above, is included in the quote below, which teaches that computer system 100 of Kelkar Fig. 1 does not employ synchronization.

> One problem with system **100** described above is that storage resource definition **140**D is stored on node **110**A. If node **110**A fails, storage resource **140** cannot be used because storage resource definition **140**D is not accessible to other nodes. To make resource configuration available to another node that can resume operation of node **110**A upon failure, the invention synchronizes resource configuration data on multiple nodes in a clustering environment.

[56] While Kelkar's system 100 of Fig. 1 does not employ synchronization, Kelkar's cluster 300 of Fig. 3 does disclose the synchronization of resource configuration data on multiple nodes in a clustering environment.

> Maintaining resource configuration data **370**A and **370**B as synchronized copies is shown in the data flow of FIG. **3**. In action **3.1**, a resource configuration is changed via resource configuration manager **360**A. In action **3.2**, resource configuration manager **360**A updates resource attributes **372**A of resource configuration data **370**A. In action **3.3**, resource configuration manager **360**A notifies resource agent **314**A for the affected resource. Resource configuration manager **360**A notifies cluster manager **330**A in action **3.4**. Cluster manager **330**A reads resource configuration data **370**A and provides configuration data **305** to all nodes in the cluster in action **3.5** via cluster communication channel **215**. Configuration data **305** may include all or a portion of resource configuration data **370**A.
>
> In action **3.6**, cluster manager **330**B receives configuration data **305** indicating that resource attributes **372**A are changed. In action **3.7**, cluster manager **330**B provides configuration data **305** to resource configuration manager **360**B. In action **3.8**, resource configuration manager **360**B updates resource attributes **372**B using configuration data **305** to reflect the changes made to resource attributes **372**A. Resource configuration data **370**A and **370**B are now synchronized.

[57] What Kelkar discloses here is that a resource configuration is changed at action 3.1. Resource attributes are updated at action 3.2. Synchronization takes place in actions 3.5-3.8, after the configuration is changed. While actions 3.5-3.8 occur as close to real-time as possible, they occur after the configuration change, not during it. The reasoning in the Examiner's Answer "The fact that Kelkar works in real time shows that Kelkar transfers the snapshots while Kelkar is still updating node 110A." starts with a false premise and reaches a false conclusion. Thus, the Examiner's theory that data is modified while snapshots are transferred is repudiated. The conclusion is that the claim limitations regarding data modification and divergence during snapshot transfer are not taught by

Kelkar. Hence, the rejections for anticipation by Kelkar should be reversed.

[58] EXAMINER'S RESPONSE GROUP 2 CLAIMS 2 AND 6

Group 2

In regard the argument which states "Since the Final Action has failed to establish that Kelkar discloses the claimed combination of a data processor and a transfer processor, the rejection for anticipation of claim 2 should be reversed"

Examiner respectfully disagrees. A closer look at figure 7 shows a computer system with which Kelkar' s invention is implemented. The figure also shows two important elements, 714 Central Processor, and 718 I/O Controller. These are two different circuitries, the first one processes the data that needs instruction processing , and the second one monitors and transfers data already processed by the first one. It is clear, in contrary to what the Appellant advances, Kelkar discloses two different processors one for regular processing, and the other one for data transfer. The argument is invalid

[59] Group 2 comprises Claim 2. Claim 2 depends from Claim 1 and adds a limitation that the processing means of Claim 1 includes a "data processor" and a "transfer processor for transferring said snapshot from said volatile memory to said storage media". Claim 1 requires the processing means to be an element of the claimed "first computer", which the Examiner has mapped to node 100A.

[60] A first problem with the Examiner's statement above is that it does not establish that Kelkar discloses the claimed elements in the claimed combination as is required in a rejection for anticipation. The Examiner maps the claimed first computer to Kelkar's node 110A, but maps components of the claimed first computer to components of Kelkar's computer system 710. Because Kelkar does not disclose the claimed elements in the claimed combination, the rejection for anticipation should be reversed.

[61] A second problem with the Examiner's statement regarding Group 2 is that Kelkar does not disclose that I/O controller 718 is involved in any way in the transfer of a snapshot from volatile memory to storage media. Kelkar only mentions I/O controller 718 in an extended list of components and provides no details as to its function. In any event, Kelkar does not disclose that I/O controller 718 performs the function required by Claim 2, so the rejection of that claim should be reversed.

[62] EXAMINER'S RESPONSE GROUP 3 CLAIM 7

Group 3

In regard the argument which states "the Final Action has failed to establish that Kelkar discloses the claim 8 limitation of using a snapshot as starting state of a second instance of an application program." Examiner respectfully disagrees.

Examiner respectfully directs Appellant to (Column 6; lines 31-33) where Kelkar discloses "When a node fails and node 110B **resumes** operations previously performed on the failed node" Resuming entails, picking up right from the point where the process was left. The pick up will not be feasible unless a copy of the resource configuration updated is contained in node 110B, and a process synchronization is available. The argument is not valid.

[63] The Examiner has directed Appellant to the following passage in Kelkar.

> In the embodiment shown in FIG. **3**, the configuration change is shared by setting attributes. For example, resource agent **314**B is responsible for configuring a newly added resource or configuring resources upon startup of node **110**B. When a node fails and node **110**B resumes operations previously performed on the failed node, resource agent **314**B reads the values of resource attributes **372**B.

[64] Here Kelkar discloses that resuming is made possible by and involves resource attributes. Kelkar does not relate these attributes to snapshots. Nor does Kelkar teach that these attributes are used as a starting state for an application. The Examiner does not relate Kelkar's teachings to snapshots. So it is still true that Kelkar does not disclose the Claim 8 limitation of using a snapshot as a starting state of a second instance of an application program.

[65] EXAMINER'S RESPONSE GROUP 4 CLAIMS 8 AND 9

[66] The Examiner has no counter to the arguments in the Appeal Brief regarding Claims 8 and 9, so the rejections of these claims should be reversed for this additional reason.

[67] EXAMINER'S RESPONSE GROUP 5 CLAIM 10

Group 5

In regard the argument which states "the Final Action has failed to establish that Kelkar discloses the claim 10 limitation of executing a second instance of an application program without detection of an intervening failure."

Examiner respectfully disagrees. This time Examiner points Appellant to (Column 5; lines 34-38) and (Column 5; lines 52-55). These passages, establish that regardless whether a failure exists or not, Kelkar updates' are dynamically processed and shared through the cluster. Argument is not valid.

[68] The Examiner points to the following passage.

FIG. **3** is a diagram of a clustering environment in which storage configuration changes can be made dynamically and are communicated throughout the cluster. Nodes **110**A and **110**B form a cluster **300**, share storage resource **140**, and communicate via cluster communication channel **215**.

[69] and

Nodes **110**A and **110**B share resource configuration data, and each node has a respective copy, respectively labeled resource configuration data **370**A and resource configuration data **370**B. Each of resource configuration data **370**A and **370**B includes respective resource attributes **372**A and **372**B.

[70] As the Examiner asserts, these passages establish that in cluster 300 updates are dynamically processed and shared through the cluster. However, neither passage teaches the Claim 10 limitation of executing a second instance of an application program (that resumes from a snapshot) without detection of an intervening failure.

[71] EXAMINER'S RESPONSE GROUP 6 AND 7 CLAIMS 11 AND 12

[72] The Examiner has no counter to the arguments in the Appeal Brief regarding Claims 11 and 12, so their rejections should be reversed for this additional reason.

[73] CONCLUSION

[74] All claims require data that continues to be modified as a snapshot of the data is transferred. As the Examiner has effectively conceded, Kelkar does not teach this limitation. To compensate for this omission, the Examiner has developed a theory that providing for synchronization "in or near real time" implies that snapshots are transferred while configurations are changed. However, Kelkar explicitly teaches (actions 3.1-3.8) that synchronization occurs after, not during, a configuration change, disconfirming the Examiner's theory and repudiating the arguments for the rejections for anticipation by Kelkar. Accordingly, all rejections of claims for anticipation by Kelkar should be reversed.